PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 :
G06F 9/445, 12/02, 12/14, 15/17, 13/00

A1

(11) International Publication Number: WO 98/19234

(43) International Publication Date: 7 May 1998 (07.05.98)

(21) International Application Number: PCT/US96/17302

(22) International Filing Date: 28 October 1996 (28.10.96)

(71) Applicant *(for all designated States except US)*: MACRONIX INTERNATIONAL CO., LTD. [-/-]; No. 3, Creation Road 3rd, Hsinchu (TW).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: SUN, Albert, C. [US/-]; 4F, No. 40, Alley 69, Lane 285, Section 1, Neihu Road, Neihu, Taipei (TW). CHEN, Chang-Lun [-/-]; 6F, No. 56, Alley 18, Lane 210, Dong-Nam Street, Hsinchu (TW). LEE, Chee-Horng [US/-]; 2F, No. 293, Chung-Chen N. Road, San-Chung, Taipei (TW).

(74) Agent: HAYNES, Mark, A.; Wilson Sonsini Goodrich & Rosati, 650 Page Mill Road, Palo Alto, CA 94304-1050 (US).

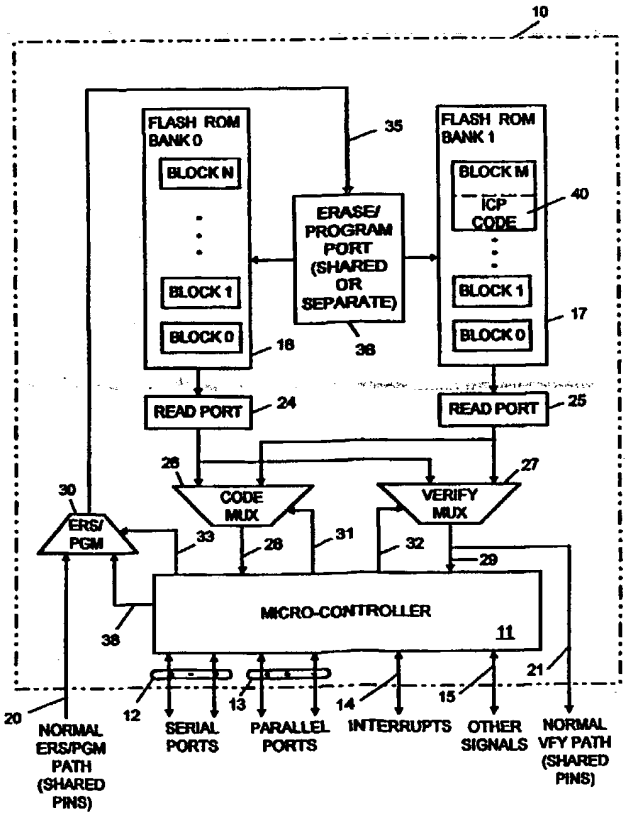
(81) Designated States: JP, US, European patent (AT, BE, CH, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).

Published
With international search report.
With amended claims.

(54) Title: PROCESSOR WITH EMBEDDED IN-CIRCUIT PROGRAMMING STRUCTURES

(57) Abstract

An architecture for an integrated circuit (10) with in-circuit programming allows for dynamically altering the in-circuit programming instruction set itself, as well as other software stored on the chip. The architecture is based on a microcontroller (11) on an integrated circuit having two or more banks of embedded non-volatile memory arrays which store instructions, including an in-circuit programming instruction set (40). Using a control program stored on the device, the device interactively establishes an in-circuit programming exchange with a remote partner, and updates data and software, including the in-circuit programming instruction set, when needed. The processor and ICP code are responsive to an in-circuit programming update command to write a copy of the in-circuit programming set from a first memory array (17) to a second memory array (16), and to cause the in-circuit programming to execute the in-circuit programming set from the second memory array to program the non-volatile memory cells of the first memory array with data from the external port.



FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	**ES**	Spain	**LS**	Lesotho	**SI**	Slovenia
AM	Armenia	**FI**	Finland	**LT**	Lithuania	**SK**	Slovakia
AT	Austria	**FR**	France	**LU**	Luxembourg	**SN**	Senegal
AU	Australia	**GA**	Gabon	**LV**	Latvia	**SZ**	Swaziland
AZ	Azerbaijan	**GB**	United Kingdom	**MC**	Monaco	**TD**	Chad
BA	Bosnia and Herzegovina	**GE**	Georgia	**MD**	Republic of Moldova	**TG**	Togo
BB	Barbados	**GH**	Ghana	**MG**	Madagascar	**TJ**	Tajikistan
BE	Belgium	**GN**	Guinea	**MK**	The former Yugoslav Republic of Macedonia	**TM**	Turkmenistan
BF	Burkina Faso	**GR**	Greece			**TR**	Turkey
BG	Bulgaria	**HU**	Hungary	**ML**	Mali	**TT**	Trinidad and Tobago
BJ	Benin	**IE**	Ireland	**MN**	Mongolia	**UA**	Ukraine
BR	Brazil	**IL**	Israel	**MR**	Mauritania	**UG**	Uganda
BY	Belarus	**IS**	Iceland	**MW**	Malawi	**US**	United States of America
CA	Canada	**IT**	Italy	**MX**	Mexico	**UZ**	Uzbekistan
CF	Central African Republic	**JP**	Japan	**NE**	Niger	**VN**	Viet Nam
CG	Congo	**KE**	Kenya	**NL**	Netherlands	**YU**	Yugoslavia
CH	Switzerland	**KG**	Kyrgyzstan	**NO**	Norway	**ZW**	Zimbabwe
CI	Côte d'Ivoire	**KP**	Democratic People's Republic of Korea	**NZ**	New Zealand		
CM	Cameroon			**PL**	Poland		
CN	China	**KR**	Republic of Korea	**PT**	Portugal		
CU	Cuba	**KZ**	Kazakstan	**RO**	Romania		
CZ	Czech Republic	**LC**	Saint Lucia	**RU**	Russian Federation		
DE	Germany	**LI**	Liechtenstein	**SD**	Sudan		
DK	Denmark	**LK**	Sri Lanka	**SE**	Sweden		
EE	Estonia	**LR**	Liberia	**SG**	Singapore		

PROCESSOR WITH EMBEDDED IN-CIRCUIT PROGRAMMING STRUCTURES

BACKGROUND OF THE INVENTION

Field of the Invention

The present invention relates to integrated circuits having a non-volatile memory for storing sequences of instructions for execution by a processor on the integrated circuit; and more to particularly to techniques for accomplishing in-circuit programming to update and modify the stored sequences of instructions.

Description of Related Art

Integrated circuit microcontrollers have been developed which include arrays of non-volatile memory on the chip for storing sequences of instructions to be executed by the microcontroller. The sequences of instructions can be stored in read only memory (ROM) which must be programmed during manufacturing of the device, and cannot be updated. In an alternative approach, the instructions can be stored in an EPROM array. These types of devices require a special programming device to be used to program the EPROM array before the device is placed in the circuit. In yet other systems, EEPROM memory can be used for storing the instructions. EEPROM can be programmed much more quickly than EPROM, and can be modified on the fly. Also, flash memory can be utilized, which allows for higher density and higher speed reprogramming of the non-volatile memory storing instructions. When the non-volatile memory is reprogrammable, such as EEPROM or flash memory, and it is combined with a microcontroller, the reprogramming of the device can be accomplished while the device is in the circuit, allowing in-circuit programming based on interactive algorithms.

For example, in the Internet environment, the ability to interactively download instruction sets and data to a remote device can be very valuable. For

example, a company can provide service to customers without requiring the customer to bring the product back to a service center. Rather, the company can execute diagnostic functions using the in-circuit programming capability of the customer's device, across a communication channel, such as the Internet or telephone lines. Software fixes can be downloaded to the customer, and the product can be reenabled with corrected or updated code.

Example prior devices which include this capability include the AT89S8252 microcontroller, manufactured by Atmel of San Jose, California, and the P83CE558 single chip microcontroller, manufactured by Philips Semiconductors of Eindhoven, The Netherlands. According to the architecture of the Philips P83CE558 microcontroller, mask ROM is utilized for the in-circuit programming ICP set of instructions, which are used for updating flash memory on the chip. Thus, the Philips microcontroller requires a dedicated mask ROM module to store fixed ICP code for each individual environment. In order to adapt the ICP code for a particular environment, the environment must be known before manufacturing is complete of the device so the mask ROM can be coded. Furthermore, the ICP communication channel is fixed to a serial RS232 port in the Philips microcontroller. The limitation to a serial port limits the application of the microcontroller to relatively narrow range of potential applications, and makes it difficult to utilize the ICP function in a dynamic communication environment where the serial port may not match well with the communication channel on which the updated software is provided.

According to the architecture of the Atmel AT89S8252 microcontroller, a dedicated Serial Peripheral Interface (SPI) port on the chip is used for the updating of flash memory storing dynamically alterable instructions. Program logic is associated with the SPI port, and therefore is inflexible. Furthermore, modification of the in-circuit programming technique cannot be done because of the hardware dedication to the SPI port. Further disadvantages of the Atmel chip include that complicated hardware used for handshaking with the ICP initiator and emulating the erase/program/verify waveform for the flash memory

must be added to the chip; that the SPI bus limitation is not always the best choice in diverse system applications, and that extra system logic is required to modify original reset circuits, which are implicated by the in-circuit programming algorithm. Further, complex SPI driver and receiver logic is required outside the chip in the system using the Atmel microcontroller.

Accordingly in-circuit programming structures have been developed which rely on flash memory or other dynamically alterable non-volatile memory. However, prior art approaches have been inflexible in the in-circuit algorithms used. Thus, in dynamic networking environments where communication requirements can change, and applications of devices using the in-circuit programming can proliferate through a wide variety of circumstances, it's desirable to provide more flexible in-circuit programming capability. Furthermore, the in-circuit programming capability must insure that no instructions are lost during the in-circuit programming process, even if the power is turned off during the process. The technique must allow for interactive communication with a remote partner to accomplish the in-circuit programming process. These techniques must be available over a wide variety of media, including the Intel/Microsoft/Digital standard Universal Serial Bus (USB), the Philips Electronics /Computer Access Technology standard Access Bus, the Apple Computer/IBM/AT&T standard Geoport, the Apple Computer/Texas Instruments/NCR standard 1349 FireWire, the Internet, a serial port (such as RS232), and other environments.

Thus, it is desirable to provide more flexible in-circuit programming structures for use with integrated circuits.

SUMMARY OF THE INVENTION

The present invention provides an architecture for an integrated circuit supporting in-circuit programming, which allows for dynamically altering the in-circuit programming instruction set itself, as well as other software stored on the chip. This greatly expands the usability of in-circuit programming devices to a

wide variety of communication environments, supporting serial ports, parallel ports, telephone communications, Internet communications, and other communication protocols as suits the needs of a particular system. The invention is based on a microcontroller or other instruction executing processor on an integrated circuit having an embedded non-volatile memory array which stores instructions, including an in-circuit programming instruction set. Using a control program stored on the device, the device interactively establishes an in-circuit programming exchange with a remote partner, and updates data and software, including the in-circuit programming sequence, when needed. For example, a self-updating algorithm, according to the present invention, proceeds as follows:

1. The device receives an in-circuit programming request.
2. The initiator is identified.
3. The device returns an identifier back to the initiator of the request.
4. The information exchange proceeds when the parties have been successfully identified.
5. In order to begin the in-circuit programming, execution of the program jumps to an in-circuit programming routine in the non-volatile memory on the chip.
6. The in-circuit programming routine mirrors itself or at least an ICP kernel, to another bank of memory on the chip.
7. The in-circuit programming software disables itself and wakes up using the mirrored ICP sequence.
8. The mirrored ICP sequence downloads new in-circuit programming software to the original ICP location.
9. The mirrored ICP sequence disables itself, and wakes up with the new ICP software in the original flash array.
10. Normal program execution is then resumed.

New instructions and data for other parts of software stored in the flash memory device can be executed at numerous places in the sequences, relying on

the original ICP code, the mirrored ICP code, or the new ICP code after it has been activated.

Accordingly, the present invention can be characterized as an apparatus for in-circuit programming of an integrated circuit having a processor which executes a program of instructions. The apparatus comprises a first memory array, comprising non-volatile memory cells, on the integrated circuit which stores instructions for execution by the processor, including in-circuit programming set of instructions. A second memory array also comprising non-volatile memory cells in preferred systems, is provided on the integrated circuit. One or more external ports is provided on the integrated circuit by which data is received from an external source. Control logic, including the processor and a kernel in the ICP code, is responsive to an in-circuit programming update command to write a copy of the in-circuit programming set from the first memory array to the second memory array, and to cause the processor to execute the in-circuit programming set from the second memory array to program the non-volatile memory cells of the first memory array with data from the external port.

According to one aspect of the invention, the first memory array comprises a plurality of separately erasable blocks of non-volatile memory cells and the in-circuit programming set is stored in a particular block. During execution of the in-circuit programming set, the particular block is modified to generate a new in-circuit programming set. Then the control logic causes the processor to execute the new in-circuit programming set from the first memory array.

According to yet another aspect of the invention, the integrated circuit includes a plurality of ports to external data sources, such as one or more serial ports, one or more parallel ports, and potentially one or more specialized communication ports. The port in the plurality of ports, as used for the external port during execution of the in-circuit programming set, is determined by the

instructions in the in-circuit programming set itself, and thus, can be dynamically altered in one preferred embodiment of the present invention.

According to another aspect of the invention, the integrated circuit includes a data path for programming and verifying the first memory array and optionally, the second memory array, independent of the in-circuit programming set of instructions. Thus, using multiplexed I/O pins or the like, original software can be loaded onto the device during manufacture or prior to mounting the chip into the system. In order to modify the original code, the in-circuit programming process is utilized.

The present invention can also be characterized as a controller on an integrated circuit that includes a processor on the integrated circuit which executes instructions received at an instruction input to the processor module. First and second memory arrays of non-volatile memory cells are provided on the integrated circuit. The first memory array stores the in-circuit programming set of instructions in a particular block of non-volatile cells. An external port is provided on the integrated circuit by which data is received from an external source. Control logic, including the processor and a kernel in the ICP code, is responsive to an in-circuit programming update command to write a copy of the in-circuit programming set from the first memory array to the second memory array, and to cause the in-circuit programming set from the second memory to program the non-volatile memory cells of the first memory array with data received from external port. Data paths on the integrated circuit are provided for programming and verifying the first memory array, independent of the in-circuit programming set. The external port comprises in alternative embodiments, a serial port and a parallel port. In yet another embodiment, there are a plurality of external ports on the device, and the one selected for use during the in-circuit programming sequence is specified by the in-circuit programming software itself.

Accordingly, a method for in-circuit programming of an integrated circuit having a processor which executes a program of instructions is provided. The method includes:

1. providing on the integrated circuit a first erasable and programmable read only memory array and a second erasable and programmable read only memory array;

2. storing an in-circuit programming set of instructions in the first array;

3. receiving an in-circuit program command from an initiator external to the integrated circuit;

4. in response to the in-circuit program command, copying the in-circuit programming set from the first array to the second array, and executing with the processor the in-circuit programming set from the second array;

5. programming at least a selected portion of the first array with data from an external source under control of the in-circuit programming set; and

6. after programming the portion of the first array, executing with the processor the in-circuit programming set from the first array.

According to another aspect of the present invention, the method includes storing the in-circuit programming set in the selected portion of the first memory array. Alternatively, the method can include the step of determining from the initiator whether the in-circuit programming sequence, indicated by the in-circuit programming command, includes modifying the in-circuit programming set. If the sequence does include the modification, then the step of copying and executing is carried out. If not, then the step of copying and executing is skipped, and the algorithm proceeds directly to programming the instructions and data in other portions of the memory array.

In sum, the present invention provides an in-circuit programming technique which allows for dynamic alteration of the in-circuit programming sequences of instructions. This enables the use of the device in a wide variety of environments, and in dynamically changing environments. For example, if a communication protocol is updated, then the in-circuit programming sequence of

instructions itself may need modification. According to the present invention, a microcontroller can be placed in the field and dynamically updated as communication protocols are improved or speeds are increased. Furthermore, the device can be adapted for a wide variety of communications ports, allowing more widespread application of the microcontroller with in-circuit programming capability.

Other aspects and advantages of the present invention can be seen upon review of the figures, the detailed description, and the claims that follow.

BRIEF DESCRIPTION OF THE FIGURES

Fig. 1 is a schematic block diagram of an integrated circuit microcontroller, including the in-circuit programming structures of the present invention.

Figs. 2A - 2C are a flow chart of a preferred in-circuit programming process, according to the present invention.

Fig. 3 is a flow chart of the update block routine which is called in the process of Figs. 2A - 2C.

Fig. 4 illustrates an environment of application of the present invention.

DETAILED DESCRIPTION

A detailed description of preferred embodiments of the present invention is provided with reference to Figs. 1, 2A - 2C, 3, and 4, in which Fig. 1 provides a hardware block diagram of the device.

Fig. 1 is a simplified diagram of an integrated circuit 10 having a processor, such as a microcontroller 11, embedded on the integrated circuit 10. Microcontroller 11 includes a plurality of serial ports 12, a plurality of parallel ports 13, interrupt pins 14, and other signal pins 15. For example, microcontoller 11 may be compatible with the standard 8051 microcontroller instruction set known in the art. A plurality of ports 12 and 13 are implemented

on the device using standard interface technology. Alternative designs include special purpose ports on chip.

To support the in-circuit programming functionality of the present invention, a first array 16 of non-volatile memory and a second array 17 of non-volatile memory are embedded on integrated circuit 10. According to a preferred embodiment, first array 16 comprises flash ROM cells, which are implemented using floating gate memory cells, as known in the art. The first array includes a plurality of blocks of flash ROM cells, block 0 through block N, and is referred to as bank 0. Second array 17 also comprises flash ROM, and includes a plurality of blocks of cells, block 0 through block M and is referred to as bank 1.

The integrated circuit includes a normal path 20 for erase and programming of arrays 16 and 17 and a normal verify path 21 for verifying the erase and programming processes, as known in the art. In a preferred system, normal paths 20 and 21 are coupled to external circuits by I/O pins on integrated circuit 10. Furthermore, these I/O pins on integrated circuit 10 are multiplexed with other pins supporting ports 12, 13, interrupts 14, and other signals 15. Thus, for example, during a test mode or manufacturing mode, the normal erase and program path 20 and the normal verify path 21 are enabled, while other functions of the I/O chips are disabled. Techniques for accomplishing the multiplexed I/O pins are common in the art.

First array 16 and second array 17 include respective read ports 24 and 25, for providing instructions stored in the arrays to microcontroller 11. Thus, read port 24 is coupled to a code input multiplexer 26, and read port 25 is coupled to the code input multiplexer 26. In a similar fashion, both read ports 24 and 25 are coupled to a verify multiplexer 27 by which data in array 16 and array 17 is read during a verify procedure.

The output of the code multiplexer 26 is supplied to an instruction input 28 for microcontroller 11. The output of the verify multiplexer 27 is supplied to a read input 29 of microcontroller 11 and to normal verify path 21.

Normal erase/program path 20 is connected to an erase/program selector 30. A second input to selector 30 includes erase and program signals on line 38 from microcontroller 11. Microcontroller 11 controls multiplexer 26 and multiplexer 27 as indicated by lines 31 and 32. Also, microcontroller 11 controls selector 30, as indicated by line 33.

Erase/program selector 30 supplies the erase and program data and signals on line 35 to erase/program port 36 for first array 16 and second array 17. The erase/program port 36 includes logic and circuits used in the erase and program operation for the arrays, such as a control state machine, a high voltage generator, a negative voltage generator, timers and the like. In the preferred embodiment, a shared erase/program circuits are used for both first array 16 and second array 17. In an alternative, separate erase/program circuits are used for the separate non-volatile memory cells. The use of separate erase/program circuits may be preferred to simplify implementation of the device, at the cost of consuming chip area.

In the embodiment illustrated in Fig. 1, only two banks of flash ROM cells are illustrated. Alternative systems include more than two banks of flash ROM cells, allowing even greater flexibility in the design and implementation of in-circuit programming structures.

As illustrated in Fig. 1, memory arrays 16 and 17 store instructions which are executed by microcontroller 11. One portion of the instructions comprises an ICP kernel, referred to as an in-circuit programming set of instructions 40. The ICP kernel includes essential ICP code and/or system code that must be preserved during in-circuit programming processes. In the embodiment shown in Fig. 1, the in-circuit programming set is stored in block M of bank 1 of flash ROM array 17. In-circuit programming set 40 can be stored at any particular block in the device in any given implementation. Also, arrays 16 and 17 include a plurality of blocks of cells which are separately erasable, according to a segmented flash architecture, for example, such as that described in United States Patent No. 5,526,307, entitled FLASH EPROM INTEGRATED

CIRCUIT ARCHITECTURE, invented by Yin, et al. Alternative embodiments include a plurality of separately erasable blocks in first array 16, and a single block in second array 17. Alternatively, a single block may be stored in each array. A wide variety of combinations of memory architectures can be implemented as suits the needs of a particular system.

Using the architecture of Fig. 1, designers of systems are enabled to adapt the in-circuit programming code to their particular environment. Thus, a manufacturer selects an integrated circuit, shown in Fig. 1, for implementation in their circuit. If the in-circuit programming code is not ready, the microcontroller is utilized and the variety of communication ports available on the chip are taken advantage of to minimize the extra logic needed on the circuit board with the microcontroller, to match the system to the particular in-circuit programming (ICP) environment envisioned. The proper connection and protocol for in-circuit programming is selected by the designer. The ICP code for the selected environment is developed and improved. Next, the ICP code is integrated with programs to be executed during normal operation of the system. Using normal erase and program path 20, the integrated ICP code and user code are stored in the flash memory arrays 16 and 17. Next, using normal verify path 21, the erase and program operations are verified. The smart microcontroller, having the integrated ICP code is then placed inside the system. The ICP procedure is executed and tested. If the ICP code works well, then the system is tested. If the system works well, then the system products can be mass produced. If the ICP code needs modification, then the process can be iteratively executed to optimize the ICP code. Similarly, the system code is optimized using the same techniques. The end user of the system developed by the manufacturer thus, has a robust in-circuit programming code embedded in the microcontroller, which can be updated and modified on the fly using the interactive in-circuit programming techniques, according to the present invention.

Figs. 2A - 2C illustrate the in-circuit programming techniques executed by end users, according to the present invention. The logic, according to Figs.

2A - 2C, is implemented by software executed by the microcontroller, by dedicated logic circuits, or by a combination of software and dedicated logic circuits.

The process begins in Fig. 2A when a remote initiator desires to update or replace the in-circuit programming set of instructions or other software stored in one of the flash ROM banks of the integrated circuit, as indicated at point 200. A first step involves the initiator sending an ICP modify request via an I/O port on the integrated circuit (block 201). The microcontroller checks the identifier of the initiator, which is carried with the request. (block 202). If the identifier check fails, the algorithm determines whether a threshold number of failures has occurred (block 203). If the test has failed too many times, then the microcontroller issues an identification fail response to the initiator (block 204), and branches to point C in Fig. 2C. If the test of block 202 has not failed too many times, then the microcontroller issues an initiator indentifier request to the initiator (block 205), and branches to block 201. If the initiator identifier test of block 202 passes, the initiator or the microcontroller issues a revision number to identify the update (block 206). Next, the microcontroller recognizes and verifies the ICP modify request in the next step (block 207). For example, in one embodiment, the microcontroller issues an ICP identification command back to the ICP initiator. The initiator responds with an acknowledgment to open an ICP communication channel.

In the next step, the microcontroller determines whether the ICP modify request has been verified (block 208). If the request is not verified, then the microcontroller continues with normal activity (block 209) and the sequence ends (block 210). If the ICP modify request is verified, then the microcontroller executes a trap to the ICP code (block 211). Thus, the microcontroller holds its current status, shuts down normal activities, and jumps to the ICP code, to begin getting ready for an in-circuit programming sequence. The process continues to point A in Fig. 2B.

The first step in Fig. 2B involves a handshake protocol with the initiator to determine the scope of the ICP modify sequence (block 212). For example, the ICP modify sequence may involve one block or many blocks in the first and second arrays in the integrated circuit. Also, the handshake protocol determines whether the ICP code itself is subject of the modify operation. Thus, the next step determines whether the ICP block is included in the modify sequence (block 213). If the ICP block is not included in the modify sequence, then the algorithm proceeds to point B as indicated, which picks up in Fig. 2C. If the ICP modify sequence does include the ICP block, then the ICP logic chooses a block for storing mirrored ICP code (block i bank 0 in the embodiment where the ICP code is originally stored in bank 1). Upon choosing the block in which to mirror the ICP code, the chosen block is erased (block 214). Next, the original ICP code is programmed into the chosen block in bank 0 (block 215).

After the program sequence, a verify operation is executed (block 216). If the verify sequence fails, then the algorithm determines whether the program of the ICP block has failed too many times (block 217). If not, then the program of the ICP block is retried by looping to block 214. If the sequence has failed too many times, then the logic issues an ICP fail response to the initiator (block 218), and branches to point C in Fig. 2C.

If at block 216, the verify procedure passes, then the code multiplexer (e.g., multiplexer 26 of Fig. 1) is switched to select instructions from the bank in which the mirrored ICP code is stored (block 219). Mirrored ICP code is executed, and a call to the update block routine with the parameters set for block M of bank 1, is made (block 220). The update block routine is illustrated in Fig. 3, and results in updating block M of bank 1, with possibly new ICP code. After step 215, the code multiplexer is switched back to the original bank, bank 1, storing the possibly new ICP code (block 221). The algorithm then proceeds to point B of Fig. 2C.

In Fig. 2C, the next step involves selecting another block for the ICP procedure, if any (block 223). After selecting another block, the update block is

called with a parameter set at block i, bank j, indicating the selected block (block 224). The algorithm next determines whether all blocks in the in-circuit programming sequence are completed (block 225). If yes, then a checksum is calculated for all programmed blocks (block 226). If no at block 225, then the algorithm loops to block 223. The loop continues until all blocks in the determined scope of the in-circuit programming procedure have been completed. After completion of the ICP procedure, a checksum is calculated for all programmed blocks (block 226). A protocol is initiated to match the calculated checksum with the checksum provided by the initiator (block 227). If a match occurs, then the microcontroller records the revision number, and issues an ICP complete response to the initiator (block 228). Then the microcontroller returns to normal operation (block 229), and the algorithm ends (block 230). If the checksum does not match at block 227, then the algorithm determines whether the ICP sequence has failed too many times (block 231). If it has not failed too many times, then the algorithm loops to point A of Fig. 2A, to retry the ICP sequence. If the ICP sequence has failed too many times, then the algorithm issues an ICP fail response to the initiator (block 232) and the procedure ends (block 230).

The update block procedure is illustrated in Fig. 3. Thus, the update block procedure is called with a parameter set, for instance block j, bank k (block 300). The sequence sets the verify multiplexer to select bank k for the verify path (block 301). Next block j of bank k is erased using the ICP erase path (block 302). After the erase process, a verify sequence is executed (block 303). If the verify fails, then it is determined whether the erase procedure has failed too many times (block 304). If not, then the algorithm loops back to block 302 to retry the erase. If it has failed too many times, then an error is returned (block 305). After successful verify from block 303, data is retrieved from the ICP initiator (block 306). The data from the ICP initiator may be one or more bytes of data depending on the ICP protocol selected by the user.

After retrieving the data from the ICP initiator, the algorithm programs block j of bank k via the ICP program path (block 307). After the program sequence, a verify operation is executed (block 308). If the program verify fails, then it is determined whether the fail has occurred too many times (block 309). If it has failed too many times, then an error is returned (block 310). If the verify has not failed too many times, then the program is retried by looping to block 307. If at block 308 the verify succeeds, then it is determined whether the ICP sequence includes more data for programming into block j of bank k (block 311). If yes, then the algorithm loops to block 306 to retrieve the next sequence of data for programming. If no more bytes remain, then the algorithm returns (block 312).

Fig. 4 illustrates the environment of application of the present invention. The present invention is implemented on an integrated circuit 400 which is placed on a printed circuit board 401 or other system implementation. The microcontroller 400 is coupled to a plurality of integrated circuits 402, 403, 404, and/or 405, in the system in which it is utilized. Chip 405 provides a bridge to a communication channel across which ICP programming is achieved. Chip 405 may comprise a simple network port, or may include extra glue logic, to make the ICP solution transparent to existing system behavior. The character of port chip 405 will be different in different applications. Chip 405 may be coupled to diverse ICP communication channels having different levels of data rates, error rates, and complexity. For example, the communication channel 406, in one embodiment, comprises an Internet protocol. Channel 406 is coupled to an ICP initiator 407, such as a personal computer or workstation. Workstation 407 is coupled by a network or other communication channel 408 to a large scale storage 409. For example, workstation 407 may be a World Wide Web site accessed through the Internet on channel 406. Alternatively, in other systems, workstation 407 acts as the initiator across a dial-up modem link. In another alternative, communication link 406 is a communication bus in a personal computer, and the in-circuit software is loaded across the bus 406, so that the

upgrades to system 401 can be distributed to end users on floppy disks or otherwise in loaded through personal computer 407.

Accordingly, the present invention provides a smart and flexible flash memory-based microcontroller architecture which allows for diverse in-circuit programming applications. For example, televisions or video monitors, digital video disks or CD-ROMs, remote control devices, or mobile telephones may include microcontrollers with in-circuit programming structures, according to the present invention. Various sources of updated ICP code can then be loaded into the respective devices using the flexible architecture of the present invention. The single in-circuit programming architecture of the present invention can be utilized in a wide variety of applications. Very little or no glue logic is needed in order to support the in-circuit programming structures. Furthermore, the power of the microcontroller associated with the in-circuit programming can be leveraged to increase the flexibility and to customized to ICP protocol for a given environment.

The foregoing description of a preferred embodiment of the invention has been presented for purposes of illustration and description. It is not intended to be exhaustive or to limit the invention to the precise forms disclosed. Obviously, many modifications and variations will be apparent to practitioners skilled in this art. It is intended that the scope of the invention be defined by the following claims and their equivalents.

CLAIMS

What is claimed is:

1. An apparatus for in-circuit programming of an integrated circuit having a processor which executes a program of instructions, comprising:

a first memory array comprising non-volatile memory cells on the integrated circuit which stores instructions for execution by the processor, including an in-circuit programming set of instructions for programming the first memory array;

a second memory array on the integrated circuit;

an external port on the integrated circuit by which data is received from an external source; and

control logic, responsive to an in-circuit program update command, to write a copy of the in-circuit programming set from the first memory array to the second memory array, and to cause the processor to execute the in-circuit programming set from the second memory array to program non-volatile memory cells of the first memory array with data from the external port.

2. The apparatus of claim 1, wherein the non-volatile memory cells in first memory array comprise floating gate memory cells.

3. The apparatus of claim 1, wherein the second memory array comprises a non-volatile memory cells.

4. The apparatus of claim 3, wherein the non-volatile memory cells in first memory array and in the second memory array comprise floating gate memory cells.

5. The apparatus of claim 1, wherein the first memory array comprises a plurality of separately erasable blocks of non-volatile memory cells, and the in-circuit programming set is stored in a particular block in the plurality of blocks, and wherein during execution of the in-circuit programming set, the particular block is modified to generate a new in-circuit programming set, and then the control logic causes the processor to execute the new in-circuit programming set from the first memory array.

6. The apparatus of claim 1, including:

a data path for programming and verifying the first memory array independent of the in-circuit programming set.

7. The apparatus of claim 1, wherein the first memory array and the second memory array comprise flash erasable and programmable read only memory.

8. The apparatus of claim 1, wherein the control logic includes instructions executed by the processor.

9. The apparatus of claim 1, wherein the external port comprises a serial port.

10. The apparatus of claim 1, wherein the external port comprises a parallel port.

11. The apparatus of claim 1, including a plurality of ports to external set data sources, and wherein the port in the plurality of ports used for the external port during execution of the in-circuit programming set is determined by the instructions in the in-circuit programming set.

12. The apparatus of claim 1, including a plurality of ports to external data sources, and wherein the in-circuit program update command is received at one of the plurality of ports.

13. The apparatus of claim 12, wherein the port in the plurality of ports used for the external port, and the port in the plurality of ports at which the in-circuit program update command is received, are determined by instructions in the in-circuit programming set.

14. The apparatus of claim 1, wherein the non-volatile memory cells in the first memory array comprise floating gate memory cells, and the second memory array comprises floating gate memory cells, and including:

an erase and program port coupled to the first and second memory arrays, which provides for erasing and programming the first and second memory arrays.

15. The apparatus of claim 14, wherein the erase and program port includes erase and program circuits connected in common to the first and second memory arrays.

16. The apparatus of claim 14, wherein the erase and program port includes separate erase and program circuits for each of the first and second memory arrays.

17. A controller on an integrated circuit, comprising:

a processor on the integrated circuit, having an instruction input, which executes instructions received at the instruction input;

a first memory array comprising non-volatile memory cells on the integrated circuit having a read port coupled to the instruction input of the

processor, and including block of non-volatile memory cells storing an in-circuit programming set of instructions;

a second memory array comprising non-volatile memory cells on the integrated circuit having a read port coupled to the instruction input of the processor;

an external port on the integrated circuit by which data is received from an external source;

control logic, on the integrated circuit, responsive to an in-circuit program update command, to write a copy of the in-circuit programming set from the first memory array to the second memory array, and to cause the processor to execute the in-circuit programming set from the second memory array to program non-volatile memory cells of the first memory array with data received at the external port; and

data paths, on the integrated circuit, for programming and verifying the first memory array independent of the in-circuit programming set.

18. The controller of claim 17, wherein the non-volatile memory cells in first memory array and in the second memory array comprise floating gate memory cells.

19. The controller of claim 17, wherein the first memory array comprises a plurality of separately erasable blocks of non-volatile memory cells, and the in-circuit programming set is stored in a particular block in the plurality of blocks, and wherein during execution of the in-circuit programming set, the particular block is modified to generate a new in-circuit programming set, and then the control logic causes the processor to execute the new in-circuit programming set from the first memory array.

20. The controller of claim 17, wherein the first memory array and the second memory array comprise flash erasable and programmable read only memory.

21. The controller of claim 17, wherein the control logic includes instructions executed by the processor.

22. The controller of claim 17, wherein the external port comprises a serial port.

23. The controller of claim 17, wherein the external port comprises a parallel port.

24. The controller of claim 17, including a plurality of ports to external data sources, and wherein the port in the plurality of ports used for the external port during execution of the in-circuit programming set is determined by the instructions in the in-circuit programming set.

25. The controller of claim 17, wherein the non-volatile memory cells in first memory array comprise floating gate memory cells, and the second memory array comprises floating gate memory cells, and including:

an erase and program port coupled to the first and second memory arrays, which provides for erasing and programming the first and second memory arrays.

26. The controller of claim 25, wherein the erase and program port includes erase and program circuits connected in common to the first and second memory arrays.

27. The controller of claim 25, wherein the erase and program port includes separate erase and program circuits for each of the first and second memory arrays.

28. A method for in-circuit programming of an integrated circuit having a processor which executes a program of instructions, comprising:

providing on the integrated circuit a first erasable and programmable read only memory array and a second erasable and programmable read only memory array;

storing an in-circuit programming set of instructions in the first array;

receiving an in-circuit program command from an initiator external to the integrated circuit;

in response to the in-circuit program command, copying the in-circuit programming set from the first array to the second array, and executing with the processor the in-circuit programming set from the second array;

programming at least a selected portion of the first array with data from an external source under control of the in-circuit programming set; and

after programming the portion of the first array, executing with the processor the in-circuit programming set from the first array.

29. The method of claim 28, wherein the step of storing the in-circuit programming set includes storing the in-circuit programming set in the selected portion of the first array.

30. The method of claim 28, wherein the first array comprises a plurality of separately erasable blocks of cells, and the selected portion of the first array comprises a particular block in the plurality of blocks, and the step of programming the portion of the first array includes modifying the particular block to generate a new in-circuit programming set.

31. The method of claim 28, including:

after modifying the particular block, executing the new in-circuit programming set from the first memory array.

32. The method of claim 28, including:

determining from the initiator whether the in-circuit programming sequence indicated by the in-circuit programming command includes modifying the in-circuit programming set, and if not, then skipping the step of copying and executing the in-circuit programming set from the second array.

33. The method of claim 28, wherein the integrated circuit includes a plurality of ports to external sources of data, and the step of programming includes receiving data from a selected port of the plurality of ports.

34. The method of claim 33, wherein the selected port is specified by the in-circuit programming set.

35. An apparatus for in-circuit programming of an integrated circuit having a processor which executes a program of instruction, comprising:

a first memory array comprising non-volatile memory cells on the integrated circuit which stores instructions for execution by the processor, including an in-circuit programming set of instructions for programming the first and the second memory array;

a second memory array comprising non-volatile memory cells on the integrated circuit which stores instruction for execution by the processor;

a code multiplexer having a first input coupled to the first memory array, a second input coupled to a second memory array, and an output to feed the processor with instructions;

an erase/program multiplexer which supports a first erase/program path for programming and erasing at least the first memory array to load at least

the first memory array with instructions, and a second erase/program path for programming and erasing at least the first memory array under control of the in-circuit programming set of instructions;

an external port on the integrated circuit by which messages are exchanged between the integrated circuit and an external source; and

logic including the in-circuit programming set of instructions stored in the first memory array and the processor, responsive to an in-circuit program update command to update all or part of the instructions stored in the first memory array and in the second memory array.

36. The apparatus of claim 35, wherein the in-circuit programming set of instructions includes logic to:

write a copy of the original in-circuit programming set from the first memory array to the second memory array to create a mirrored in-circuit programming set;

control the code multiplexer to switch the processor control from the original in-circuit programming set to the mirrored in-circuit programming set in the second memory array; and

update the in-circuit programming set in the first memory array under the control of the mirrored in-circuit programming set in the second memory array.

37. The apparatus of claim 35, including program and erase circuits coupled in common to the first and second memory arrays.

38. The apparatus of claim 35, wherein the non-volatile memory cells in first memory array comprise floating gate memory cells.

39. The apparatus of claim 35, wherein the non-volatile memory cells in first memory array and in the second memory array comprise floating gate memory cells.

40. The apparatus of claim 35, wherein the first memory array comprises a plurality of separately erasable blocks of non-volatile memory cells.

41. The apparatus of claim 35, wherein the first memory array and the second memory array comprise flash erasable and programmable read only memory.

42. The apparatus of claim 35, wherein the external port comprises a serial port.

43. The apparatus of claim 35, wherein the external port comprises a parallel port.

44. The apparatus of claim 35, including a plurality of ports to external set data sources, and wherein the port in the plurality of ports used for the external port during execution of the in-circuit programming set is determined by the instructions in the in-circuit programming set.

45. The apparatus of claim 35, including a plurality of ports to external data sources, and wherein the in-circuit program update command is received at one of the plurality of ports.

46. The apparatus of claim 45, wherein the port in the plurality of ports used for the external port, and the port in the plurality of ports at which the in-circuit program with data command is received and determined by the instruction in the in-circuit programming.

47. A method for in-circuit programming of an integrated circuit having a processor which executes a program of instructions, comprising:

providing on the integrated circuit a first erasable and programmable read only memory array and a second erasable and programmable read only memory array;

storing an in-circuit programming set of instructions in the first array;

receiving an in-circuit program command and an identifier from an initiator external to the integrated circuit;

verifying the identifier of the initiator, and if verified then in response to the in-circuit program command, copying the in-circuit programming set from the first array to the second array, and executing with the processor the in-circuit programming set from the second array; and

programming at least a selected portion of the first array with an instruction set from an external source under control of the in-circuit programming set.

48. The method of claim 47, including:

after programming the portion of the first array, executing with the processor the in-circuit programming set from the first array.

49. The method of claim 47, including:

if the identifier of the initiator is not verified, then requesting identification from the initiator, and attempting to verify the identification of the initiator until verification succeeds, or until a maximum number of attempts has been made; and

if the maximum number of attempts is made, then issuing a fail response to the initiator.

50. The method of claim 47, including:

issuing a revision number to identify the instruction set, and storing the revision number if the step of programming succeeds.

51. The method of claim 47, including:

after the step of programming, checking the programmed instruction set for errors, and if an error is found, the retrying the programming step until no errors are found or until a maximum number of retries has been attempted; and

if the maximum number of retries is attempted, then issuing a fail response to the initiator.

52. The method of claim 51, wherein the step of checking the programmed instruction set for errors includes verifying a checksum for the programmed instruction set.

53. The method of claim 47, wherein the step of programming includes programming a segment of the instruction set, and verifying programming of the segment, and if the programming of the segment is verified, proceeding to a next segment until the instruction set is completed, and if the programming of the segment is not verified, then retrying the programming until it passes verify or until a maximum number of retries has been attempted;

if the maximum number of retries is attempted without passing verify, then issuing a fail response to the initiator.

54. The method of claim 47, wherein the step of storing the in-circuit programming set includes storing the in-circuit programming set in the selected portion of the first array.

55. The method of claim 47, wherein the first array comprises a plurality of separately erasable blocks of cells, and the selected portion of the first array comprises a particular block in the plurality of blocks, and the step of

programming the portion of the first array includes modifying the particular block to generate a new in-circuit programming set.

56. The method of claim 55, including:

after modifying the particular block, executing the new in-circuit programming set from the first memory array.

57. The method of claim 47, including:

determining from the initiator whether the in-circuit programming sequence indicated by the in-circuit programming command includes modifying the in-circuit programming set, and if not, then skipping the step of copying and executing the in-circuit programming set from the second array.

58. The method of claim 47, wherein the integrated circuit includes a plurality of ports to external sources of data, and the step of programming includes receiving data from a selected port of the plurality of ports.

59. The method of claim 58, wherein the selected port is specified by the in-circuit programming set.

AMENDED CLAIMS

[received by the International Bureau on 15 April 1997 (15.04.97); new claims 60-71 added; remaining claims unchanged (3 pages)]

programming the portion of the first array includes modifying the particular block to generate a new in-circuit programming set.

56. The method of claim 55, including:

after modifying the particular block, executing the new in-circuit programming set from the first memory array.

57. The method of claim 47, including:

determining from the initiator whether the in-circuit programming sequence indicated by the in-circuit programming command includes modifying the in-circuit programming set, and if not, then skipping the step of copying and executing the in-circuit programming set from the second array.

58. The method of claim 47, wherein the integrated circuit includes a plurality of ports to external sources of data, and the step of programming includes receiving data from a selected port of the plurality of ports.

59. The method of claim 58, wherein the selected port is specified by the in-circuit programming set.

60. An apparatus for in-circuit programming of an integrated circuit comprising:

a processor on the integrated circuit which executes instructions;

an external port on the integrated circuit through which data is received from an external source; and

a first memory array comprising reprogrammable non-volatile memory cells on the integrated circuit, which stores instructions for execution by the processor, including a set of instructions for controlling the transfer of a

set of in-circuit programming instructions into the integrated circuit from the external source through the external port.

61. The apparatus of claim 60, wherein the in-circuit programming set of instructions includes logic to:

write a copy of the original in-circuit programming set from the first memory array to the second memory array to create a mirrored in-circuit programming set;

control the code multiplexer to switch the processor control from the original in-circuit programming set to the mirrored in-circuit programming set in the second memory array; and

update the in-circuit programming set in the first memory array under the control of the mirrored in-circuit programming set in the second memory array.

62. The apparatus of claim 60, including program and erase circuits coupled in common to the first memory array.

63. The apparatus of claim 60, wherein the non-volatile memory cells in first memory array comprise floating gate memory cells.

64. The apparatus of claim 60, wherein the non-volatile memory cells in first memory array comprise floating gate memory cells.

65. The apparatus of claim 60, wherein the first memory array comprises a plurality of separately erasable blocks of non-volatile memory cells.

66. The apparatus of claim 60, wherein the first memory array comprises flash erasable and programmable read only memory.

67. The apparatus of claim 60, wherein the external port comprises a serial port.

68. The apparatus of claim 60, wherein the external port comprises a parallel port.

69. The apparatus of claim 60, including a plurality of ports to external set data sources, and wherein the port in the plurality of ports used for the external port during execution of the in-circuit programming set is determined by the instructions in the in-circuit programming set.

70. The apparatus of claim 60, including a plurality of ports to external data sources, and wherein the in-circuit program update command is received at one of the plurality of ports.

71. The apparatus of claim 70, wherein the port in the plurality of ports used for the external port, and the port in the plurality of ports at which the in-circuit program with data command is received and determined by the instruction in the in-circuit programming.

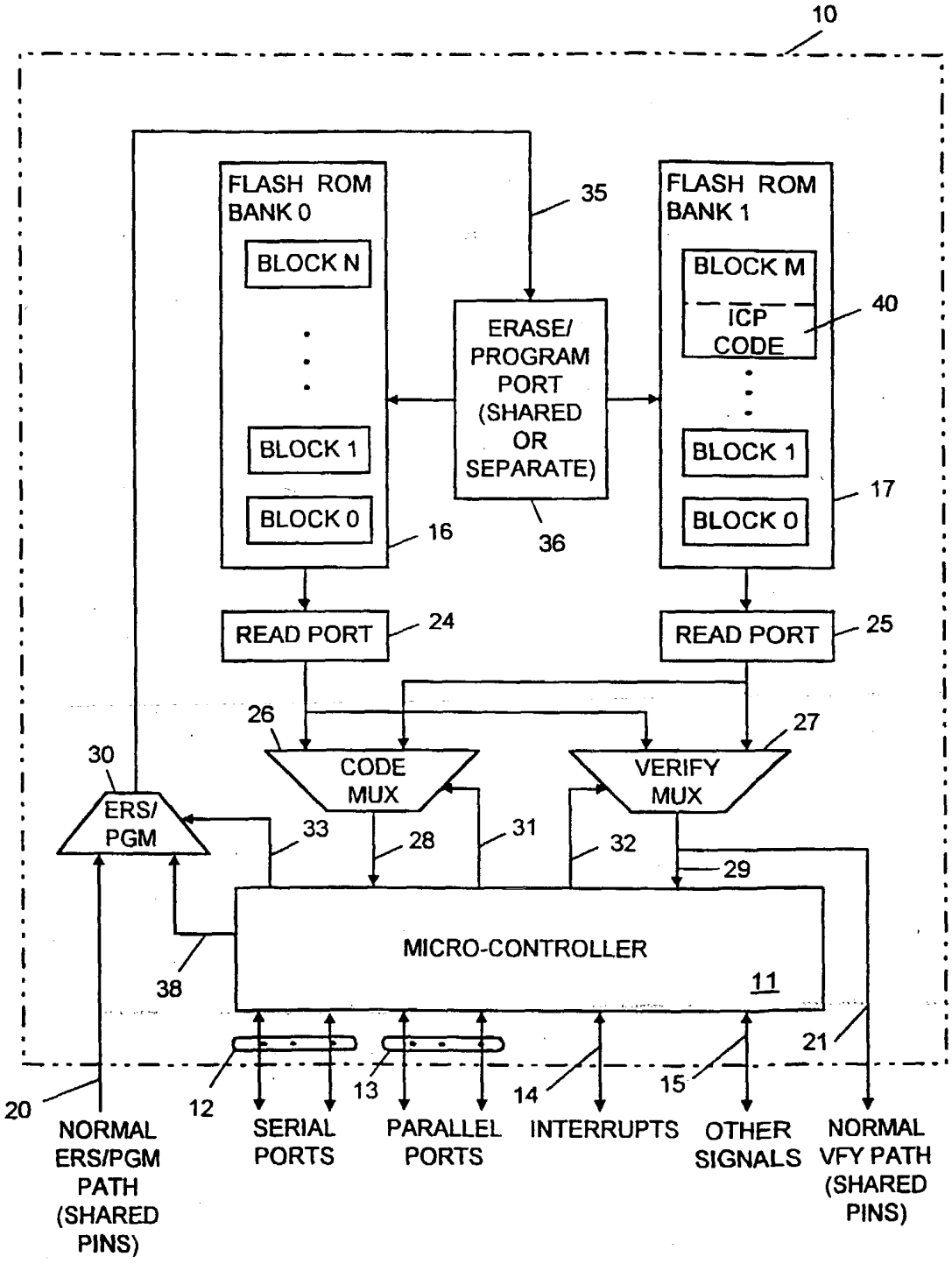

10
FLASH ROM BANK 0
BLOCK N
BLOCK 1
BLOCK 0
16
35
ERASE/ PROGRAM PORT (SHARED OR SEPARATE)
36
FLASH ROM BANK 1
BLOCK M
ICP CODE
40
BLOCK 1
BLOCK 0
17
READ PORT
24
READ PORT
25
26
27
CODE MUX
VERIFY MUX
30
ERS/ PGM
33
28
31
32
29
MICRO-CONTROLLER
11
38
21
20
12
13
14
15
NORMAL ERS/PGM PATH (SHARED PINS)
SERIAL PORTS
PARALLEL PORTS
INTERRUPTS
OTHER SIGNALS
NORMAL VFY PATH (SHARED PINS)

FIG. 1

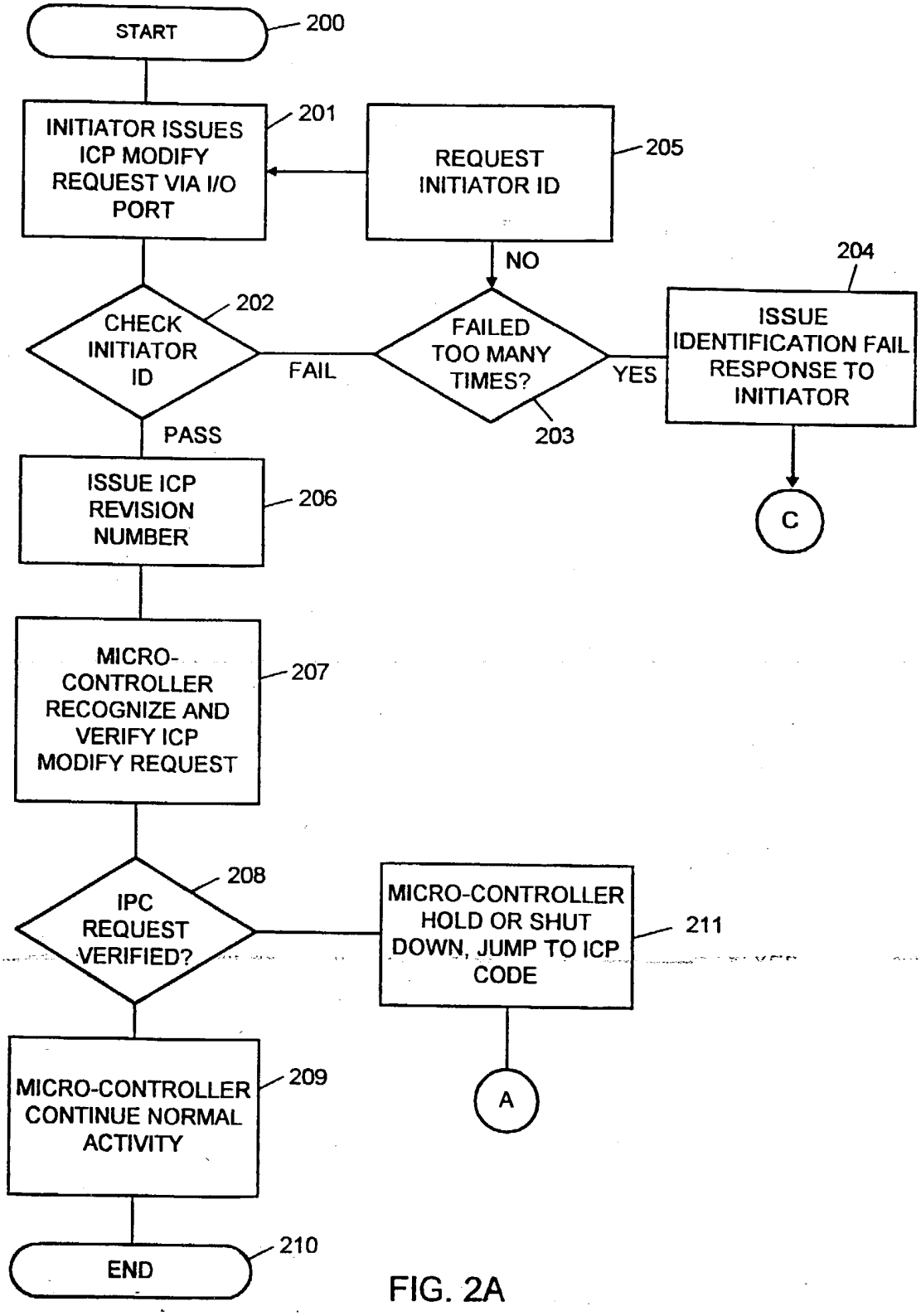

START
200
INITIATOR ISSUES ICP MODIFY REQUEST VIA I/O PORT
201
REQUEST INITIATOR ID
205
NO
204
CHECK INITIATOR ID
202
FAIL
FAILED TOO MANY TIMES?
YES
ISSUE IDENTIFICATION FAIL RESPONSE TO INITIATOR
203
PASS
C
ISSUE ICP REVISION NUMBER
206
MICRO-CONTROLLER RECOGNIZE AND VERIFY ICP MODIFY REQUEST
207
IPC REQUEST VERIFIED?
208
MICRO-CONTROLLER HOLD OR SHUT DOWN, JUMP TO ICP CODE
211
A
MICRO-CONTROLLER CONTINUE NORMAL ACTIVITY
209
END
210

FIG. 2A

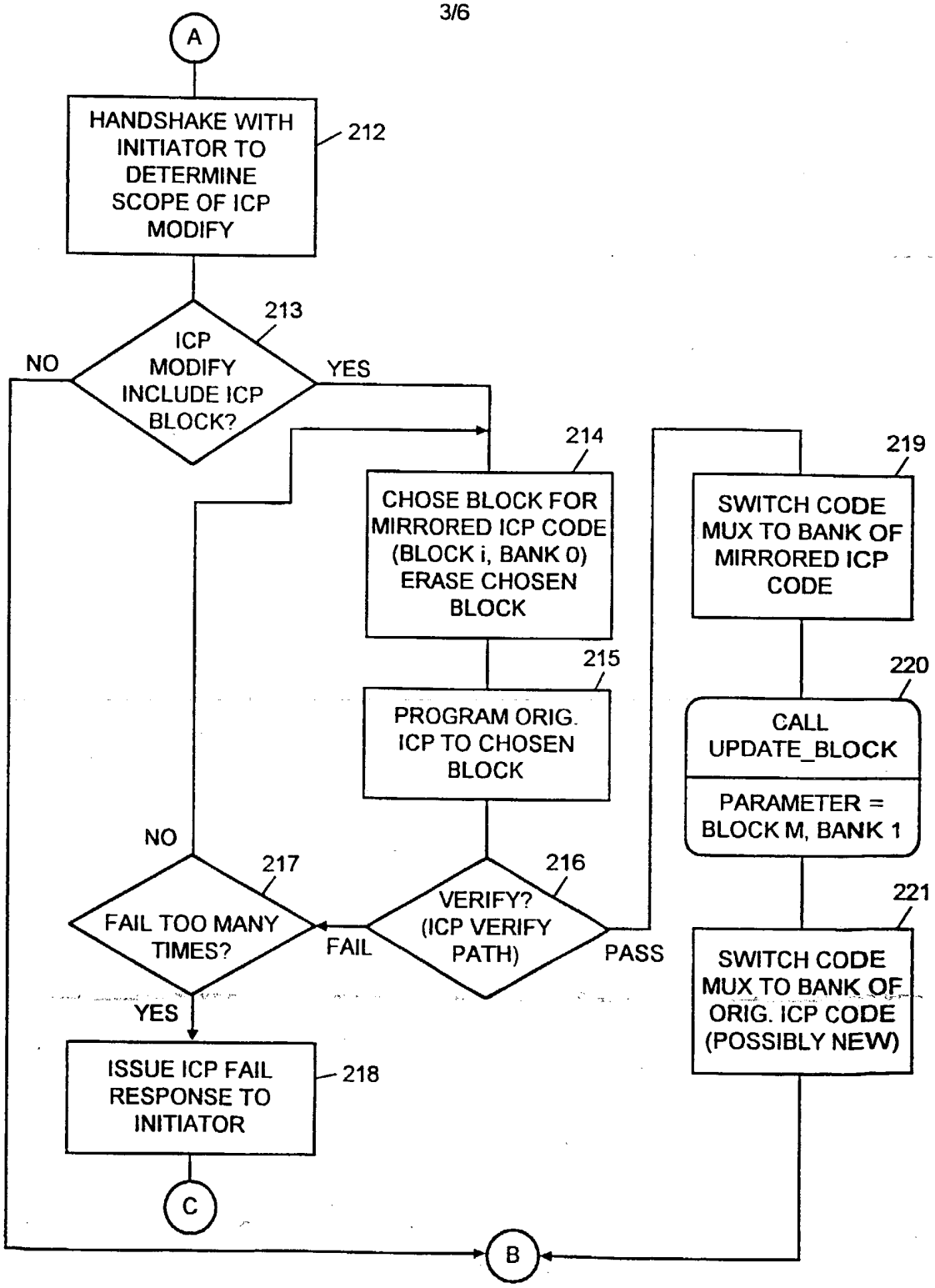

A
HANDSHAKE WITH INITIATOR TO DETERMINE SCOPE OF ICP MODIFY
212
213
ICP MODIFY INCLUDE ICP BLOCK?
NO
YES
214
CHOSE BLOCK FOR MIRRORED ICP CODE (BLOCK i, BANK 0) ERASE CHOSEN BLOCK
215
PROGRAM ORIG. ICP TO CHOSEN BLOCK
216
VERIFY? (ICP VERIFY PATH)
PASS
FAIL
217
FAIL TOO MANY TIMES?
NO
YES
218
ISSUE ICP FAIL RESPONSE TO INITIATOR
C
219
SWITCH CODE MUX TO BANK OF MIRRORED ICP CODE
220
CALL UPDATE_BLOCK
PARAMETER = BLOCK M, BANK 1
221
SWITCH CODE MUX TO BANK OF ORIG. ICP CODE (POSSIBLY NEW)
B

FIG. 2B

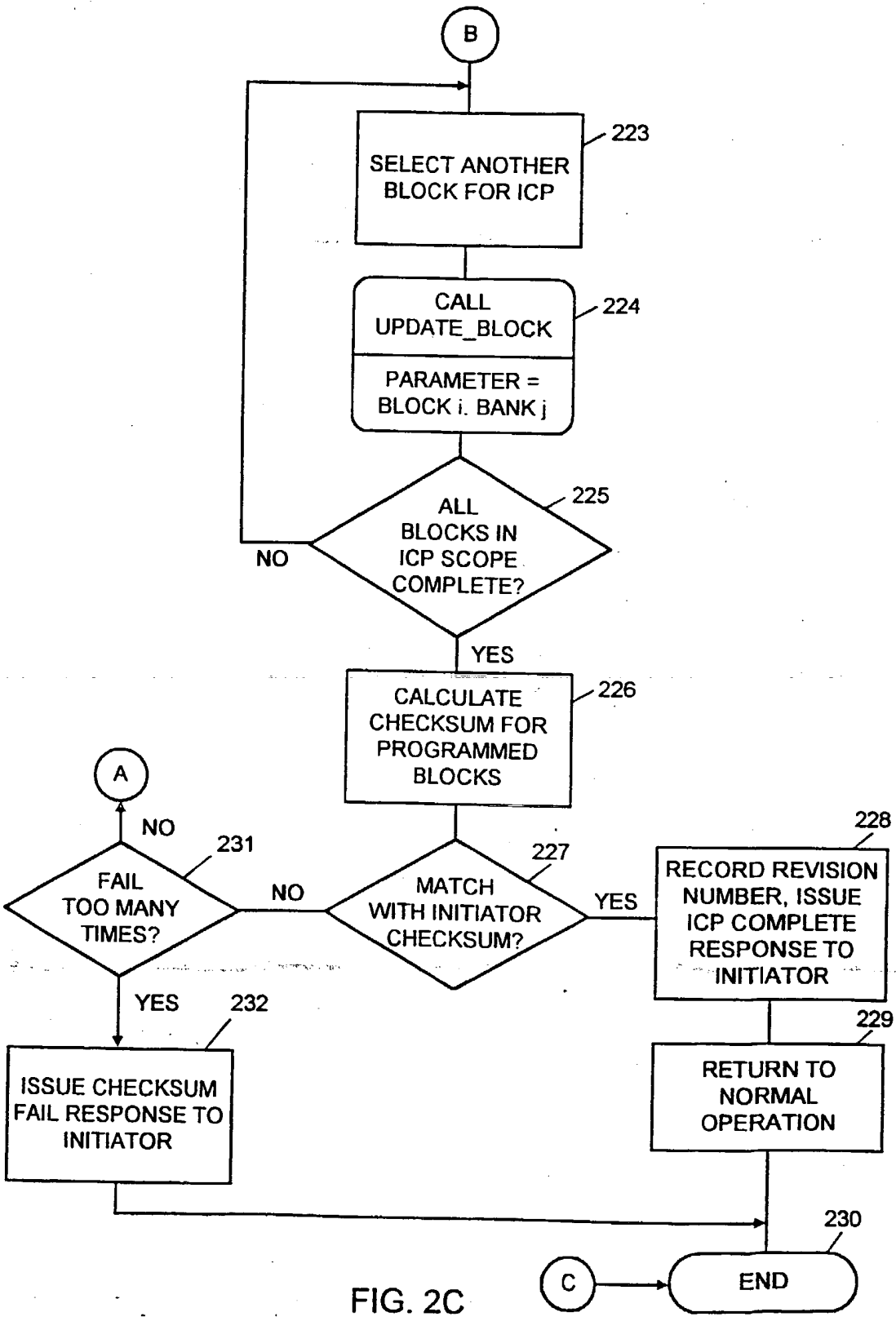

B
SELECT ANOTHER BLOCK FOR ICP
223
CALL UPDATE_BLOCK
PARAMETER = BLOCK i. BANK j
224
ALL BLOCKS IN ICP SCOPE COMPLETE?
225
NO
YES
CALCULATE CHECKSUM FOR PROGRAMMED BLOCKS
226
A
NO
FAIL TOO MANY TIMES?
231
NO
MATCH WITH INITIATOR CHECKSUM?
227
YES
228
RECORD REVISION NUMBER, ISSUE ICP COMPLETE RESPONSE TO INITIATOR
YES
232
ISSUE CHECKSUM FAIL RESPONSE TO INITIATOR
229
RETURN TO NORMAL OPERATION
230
C
END

FIG. 2C

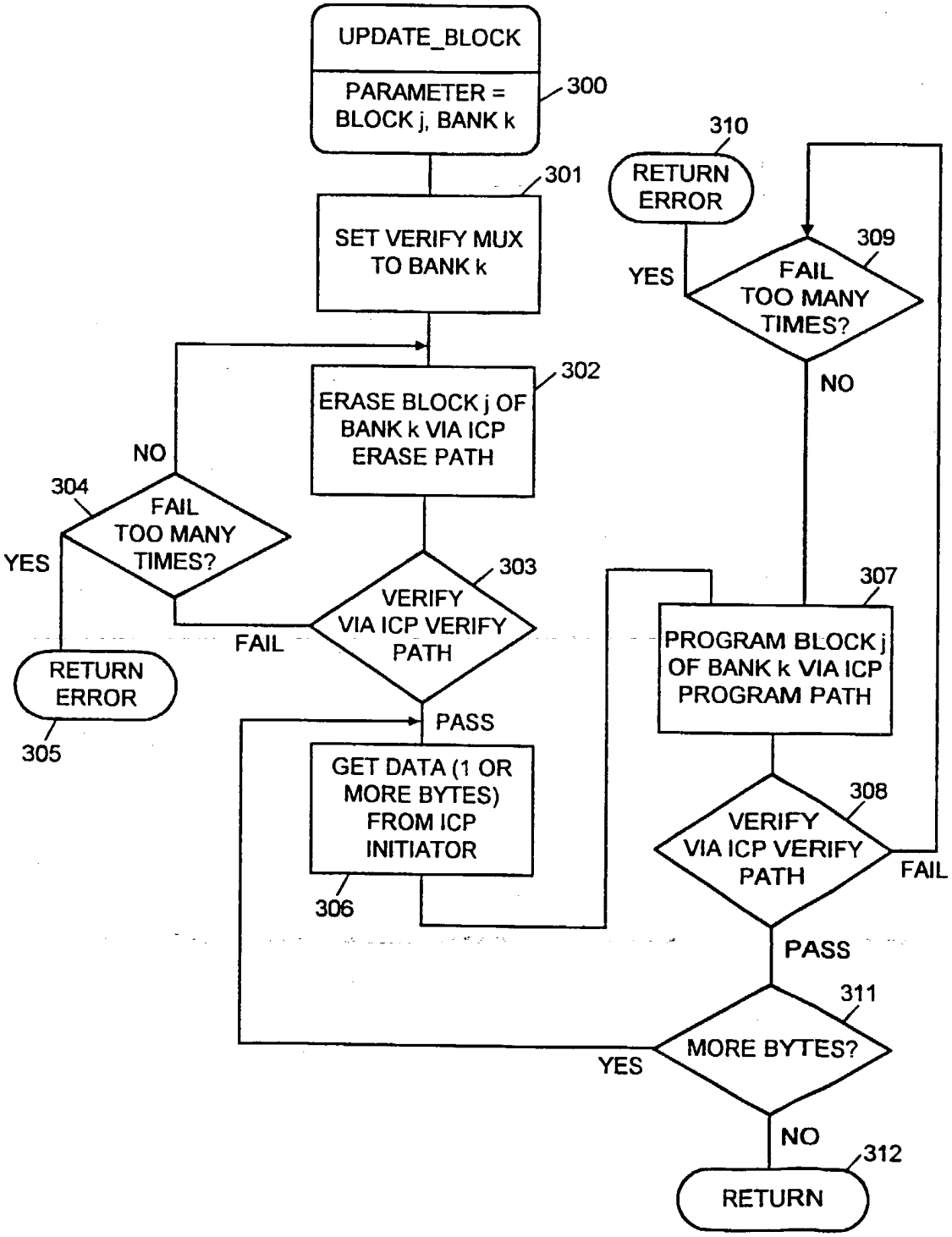

UPDATE_BLOCK
PARAMETER = BLOCK j, BANK k
300
SET VERIFY MUX TO BANK k
301
ERASE BLOCK j OF BANK k VIA ICP ERASE PATH
302
VERIFY VIA ICP VERIFY PATH
303
FAIL TOO MANY TIMES?
304
NO
YES
FAIL
RETURN ERROR
305
PASS
GET DATA (1 OR MORE BYTES) FROM ICP INITIATOR
306
PROGRAM BLOCK j OF BANK k VIA ICP PROGRAM PATH
307
VERIFY VIA ICP VERIFY PATH
308
FAIL
PASS
FAIL TOO MANY TIMES?
309
YES
NO
RETURN ERROR
310
MORE BYTES?
311
YES
NO
RETURN
312

FIG. 3

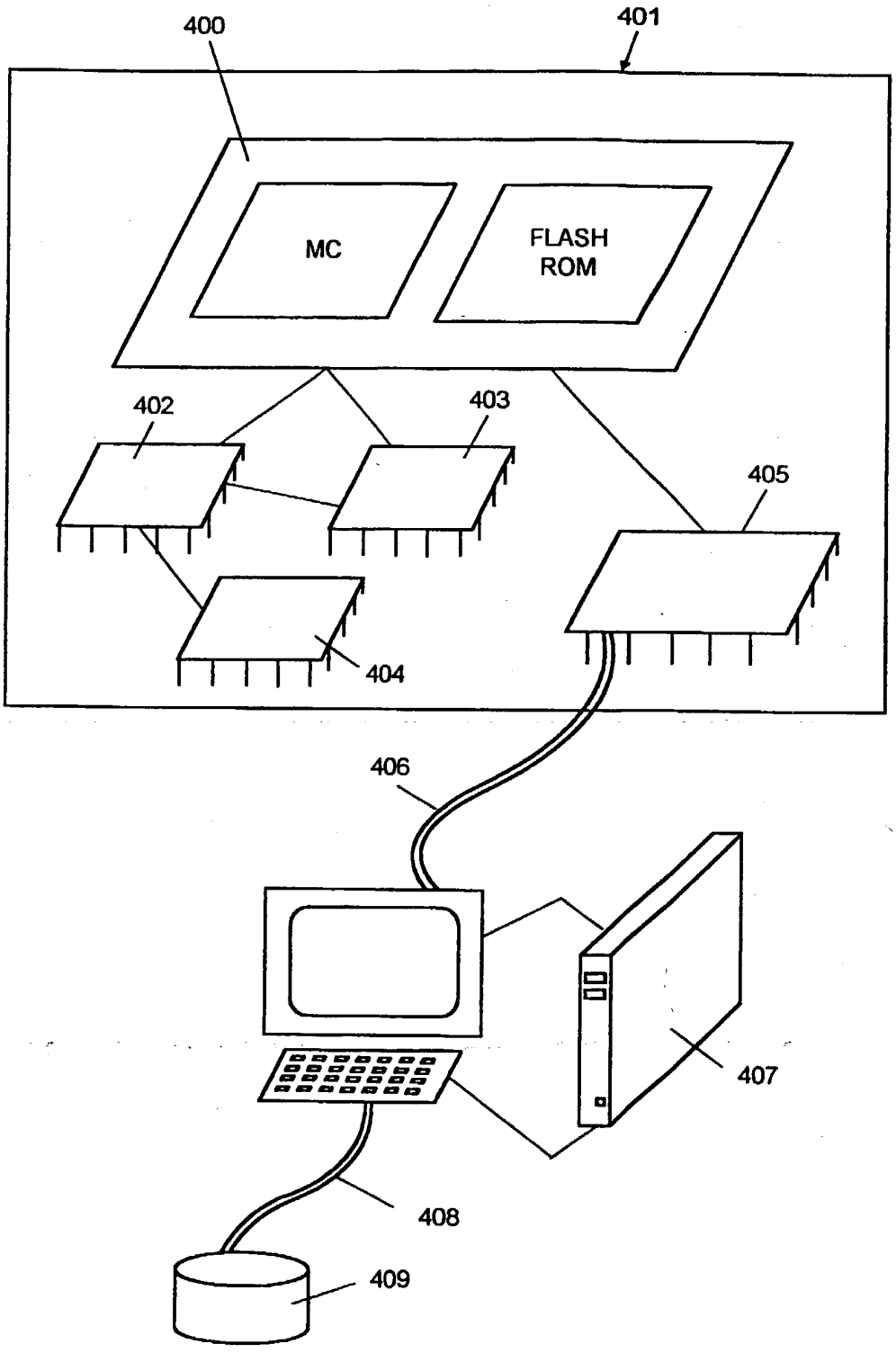


FIG. 4

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US96/17302

A. CLASSIFICATION OF SUBJECT MATTER

IPC(6) :G06F 9/445, 12/02, 12/14, 15/17, 13/00
US CL :395/492, 491, 712, 188.01
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

U.S. : 395/492, 491, 712, 188.01

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

Please See Extra Sheet.

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X Y	US 5,467,286 A (PYLE ET AL) 14 November 1995, col. 1, lines 54-66, col. 7, line 3 to col. 9, line 13.	1-2, 5, 7-8 3-6, 9-59
Y	US 5,444,861 A (ADAMEC ET AL) 22 August 1995, col. 3, lines 33-61.	6,9,17,22,42
Y,E	US 5,579,479 A (PLUM) 26 November 1996, col. 5, line 41 to col. 6, line 65.	47-59
Y	US 5,018,096 A (AOYAMA) 21 May 1991, col. 2, lines 1-13.	47-59
Y	US 5,163,147 A (ORITA) 10 November 1992, col. 1, lines 57 to col. 2, line 18, col. 3, lines 10-52.	47-59

[X] Further documents are listed in the continuation of Box C. [ ] See patent family annex.

*	Special categories of cited documents:	"T"	later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"A"	document defining the general state of the art which is not considered to be of particular relevance	"X"	document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"E"	earlier document published on or after the international filing date	"Y"	document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"L"	document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)	"&"	document member of the same patent family
"O"	document referring to an oral disclosure, use, exhibition or other means		
"P"	document published prior to the international filing date but later than the priority date claimed		

Date of the actual completion of the international search 12 DECEMBER 1996	Date of mailing of the international search report 29 JAN 1997
Name and mailing address of the ISA/US Commissioner of Patents and Trademarks Box PCT Washington, D.C. 20231 Facsimile No. (703) 305-3230	Authorized officer REGINALD G. BRAGDON Telephone No. (703) 305-9600

Form PCT/ISA/210 (second sheet)(July 1992)*

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US96/17302

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y,E	US 5,581,723 A (HASBUN ET AL) 03 December 1996, col. 4, line 3 to col. 8, line 39.	1-59

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US96/17302

B. FIELDS SEARCHED
Electronic data bases consulted (Name of data base and where practicable terms used):

Proquest (IEEE Periodicals 1988-1996), APS text, JPO abstracts, EPO abstracts, DR. Dobbs
search terms: in-circuit programming, ICP, code, instruction, nvs, non-volatile or nonvolatile, memory, flash, eeprom, rom, feeprom, automatic or autonomous update, patch, upgrade, update, revision, download, security, password, identifier, parallel port, serial port, software